Dealing with degenerated cases in quadratic programming

Eduardo Casas, Cecilia Pola

INRIA

UNITE DE RECHERCHE
INRIA-ROCQUENCOURT

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
BP 105
78153 Le Chesnay Cedex
France
Tel (1) 39 63 55 11

Rapports de Recherche

N° 1126

Programme 5
Automatique, Productique,
Traitement du Signal et des Données

DEALING WITH DEGENERATED CASES IN QUADRATIC PROGRAMMING

Eduardo CASAS
Cecilia POLA

Décembre 1989

RR-1126

TRAITEMENT DES CAS DEGENERES EN PROGRAMMATION QUADRATIQUE

DEALING WITH DEGENERATED CASES IN QUADRATIC PROGRAMMING

Eduardo Casas and Cecilia Pola

Departamento de Matemáticas, Estadística y Computación

Universidad de Cantabria

39071-Santander (SPAIN)

ABSTRACT

A new algorithm is described for quadratic programming which is based on a Cholesky factorization that uses a diagonal pivoting strategy and that allows to compute null or negative curvature directions. The algorithm is numerically stable and has shown efficiency solving positive-definite and indefinite problems. It is specially interesting in indefinite cases because the initial point does not need to be a vertex of the feasible set. So we avoid introducing artificial constraints in the problem, which turns out to be very efficient in parametric programming. At the same time techniques for updating matrix factorizations are used.

RESUME

Nous présentons dans ce rapport un nouveau algorithme de programmation quadratique qui repose sur une factorisation de Cholesky avec une stratégie de pivotation diagonale et qui permet de calculer directions de curvature nulle ou negative. L'algorithme est numeriquement stable et il a montré efficacité pour résoudre des problèmes définis positives et indéfinis. Il est notamment intéressant dans les cas indefinis parçe que le point initial n'a pas besoin d'être un extrême de la région de points admissibles. Donc nous évitons d'introduire des contraintes artificielles dans le problème, ce qui revient très agissant dans la programmation paramétrique. Au même temps nous utilisons des téchniques pour adapter les factorizations des matrices.

Une partie de ce travail a été effectuée lors d'un séjour des auteurs dans le projet PROMATH en Octobre 1989.

1. INTRODUCTION

The aim of this paper is to present a new algorithm for solving the following quadratic problem

$$
\text{(QP)} \qquad
\begin{aligned}
&\text{Minimize} && F(x) = \tfrac{1}{2}x^T H x + p^T x \\
&\text{Subject to} && c_j^T x = b_j && 1 \le j \le m_e \\
& && c_j^T x \le b_j && m_e + 1 \le j \le m_e + m_i \\
& && l_i \le x_i \le u_i && 1 \le i \le n
\end{aligned}
$$

where H is a n x n symmetric matrix, p and c_j are n-vectors, b_j are real numbers and l_i and u_i are elements of $[-\infty,+\infty]$ satisfying: $l_i \le u_i$. In the sequel we will denote by C the n x (m_e+m_i) matrix that collects the column vectors c_j and b will be the vector $(b_j)_{j=1}^{m}$, with $m = m_e + m_i$.

Although a code of quadratic programming must consider bound constraints separately from more general inequality constraints $c_j^T x \le b_j$, in order to simplify the exposition we are going to formulate the problem in the way

$$
\text{(QP)} \qquad
\begin{aligned}
&\text{Minimize} && F(x) = \tfrac{1}{2}x^T H x + p^T x \\
&\text{Subject to} && c_j^T x = b_j && 1 \le j \le m_e \\
& && c_j^T x \le b_j && m_e + 1 \le j \le m_e + m_i
\end{aligned}
$$

The most of codes solving this problem for indefinite matrices H follow a strategy for choosing certain subset of active constraints (the working set)

that ensures that the reduced Hessian respect to the working set never has more than one nonpositive eigenvalue. These methods need to start from a feasible point x^0 which is a vertex of the feasible point region or in other case it is necessary to add artificial constraints to the problem so that the initial point is at a vertex. These constraints are deleted from the working set as soon as possible which means that the algorithm must perform at least so iterations as number of artificial constraints have been added, see R. Fletcher [3], [4] and P.E. Gill et al. [7], [8]. This can retard the resolution of problem, particularly in parametric programming when the active constraints are close to be identified.

Our method allows any number of nonpositive eigenvalues in the reduced Hessian and therefore it does not need to start from a vertex of the feasible point region. Other advantage of our strategy is that if the objective function is bounded below in the feasible region and the feasible region contains no degenerate stationary points, the algorithm converges in a finite number of iterations to a local (or global) minimum. In contrast, with the strategy above mentioned we can only ensure the convergence to a Kuhn-Tucker point.

Other method allowing any number of nonpositive eigenvalues in the reduced Hessian is dued to Bunch and Kaufman [1]. Their method is based on the decomposition $Q = MDM^T$ of a symmetric matrix Q, where D is block diagonal with blocks of order 1 or 2, and M is the product of permutations and block elementary transformations. Our algorithm is based on the Cholesky decomposition.

In this paper we will see that it is possible to get feasible descent directions of negative, null or positive curvature from the Cholesky factorization of the reduced Hessian which allows us to deal with any degenerated case of quadratic programming. This is performed without introducing any artificial restrictions and starting the algorithm at any feasible point x^0. Obviously numerical stability requires to stop the factorization when indefiniteness of matrix is detected, in this case we compute a negative curvature direction from the partial factorization. When the matrix is positive semi-definite it is possible to carry out the complete factorization and to derive a null or positive curvature descent direction. We will also show that it is possible to update the Cholesky factors in all cases in a similar form to that used by P.E. Gill and W. Murray in [6].

The plan of this paper is the following. In the next section we study the Cholesky factorization of a symmetric matrix A and show the way of getting negative or null curvature directions. In Section 3 the proposed quadratic programming model algorithm is stated. Updating of matrix factors is considered in Section 4 and two numerical examples are studied in section 5.

2. SOME QUESTIONS ABOUT CHOLESKY DECOMPOSITION

Given a n x n matrix A that is symmetric but not necessarily positive definite, we are going to propose an algorithm that supplies the Cholesky decomposition of PAP^T (where P is a permutation matrix) and a basis of its

kernel when A is positive semi-definite and that realizes an incomplete factorization and furnishes a negative curvature direction when A is indefinite. We first establish the following theorem:

THEOREM 1. Matrix A is positive semi-definite if and only if there exists a permutation matrix P such that

$$PAP^T = \begin{bmatrix} L & 0 \\ B & 0 \end{bmatrix} \begin{bmatrix} L^T & B^T \\ 0 & 0 \end{bmatrix}$$

where L is a m x m lower-triangular matrix with strictly positive diagonal elements and B is a (n-m) x m matrix. m is the rank of A and a basis of its kernel is formed by the vectors $\{P^T u_j\}_{j=1}^{n-m}$ defined by the formula

$$u_j = \begin{bmatrix} \hat{u}_j \\ 0 \end{bmatrix} - e_{m+j}$$

where e_{m+j} is the (m+j)-th column of the n x n identity matrix and $\hat{u}_j$ is the m-vector solution of the system $L^T\hat{u}_j = B_j^T$, B_j being the j-th row of B.

PROOF. It is easy to verify that $\{P^T u_j\}_{j=1}^{n-m}$ is a basis of the kernel of A. On the other hand it is well known that A is positive semi-definite if and only if the above factorization is possible, see for example [2].

Now we propose an algorithm that finds out if A is positive semi-definite or indefinite and performs the Cholesky decomposition in the first case:

ALGORITHM

1) Set k=1, $A^{(k)} = (a_{ij}^{(k)}) = A$, $P^{(k)}$ = Identity and

$$\beta = 1.2 \cdot \left(\max \{ |a_{jj}|, j = 1,\ldots,n \} \right)^{1/2}$$

2) Find q such that

$$a_{qq}^{(k)} = \max \{ a_{jj}^{(k)}, j = k,\ldots,n \}$$

If $a_{qq}^{(k)} < 0 \Rightarrow$ Indefinite matrix. STOP.

If $a_{qq}^{(k)} = 0 \Rightarrow$ Find t such that

$$a_{tt}^{(k)} = \min \{ a_{jj}^{(k)}, j = k,\ldots,n \}$$

If $a_{tt}^{(k)} < 0 \Rightarrow$ Indefinite matrix. STOP.

If $a_{tt}^{(k)} = 0 \Rightarrow$ Find r and s, with $r > s$, such that

$$|a_{rs}^{(k)}| = \max \{ |a_{ij}^{(k)}|, n \geq i > j \geq k \}$$

If $|a_{rs}^{(k)}| > 0 \Rightarrow$ Indefinite matrix. STOP.

If $a_{rs}^{(k)} = 0 \Rightarrow$ End of factorization. STOP.

If $a_{qq}^{(k)} > 0 \Rightarrow$ Interchange the rows and columns q and k of $A^{(k)}$, the new matrix being denoted again by $A^{(k)}$. Perform the same interchange of rows in $P^{(k)}$ and denote the new matrix by $P^{(k+1)}$.

3) Apply the following formulas:

From j = 1 to k-1

$$a_{ij}^{(k+1)} = a_{ij}^{(k)} \qquad i = j,\ldots,n$$

$$a_{kk}^{(k+1)} = \sqrt{a_{kk}^{(k)}}$$

If k = n ⇒ End of factorization. STOP.

If k < n ⇒ Continue

From i = k+1 to n

$$a_{ik}^{(k+1)} = \frac{a_{ik}^{(k)}}{a_{kk}^{(k+1)}}$$

If max { $|a_{ik}^{(k+1)}|$, i= k+1,...,n } > β ⇒ Indefinite matrix. STOP.

From j = k+1 to n

$$a_{jj}^{(k+1)} = a_{jj}^{(k)} - \left(a_{jk}^{(k+1)} \right)^2$$

$$a_{ij}^{(k+1)} = a_{ij}^{(k)} - a_{ik}^{(k+1)} a_{jk}^{(k+1)} \qquad i= k+1,\ldots,n$$

4) Go to Step 2).

REMARKS. (1) In practice the determination of A as a positive semi-definite

matrix is based on the choice of a parameter TOL, which depends on size of A and machine precision. We have taken

$$TOL = n \cdot \max \{ |a_{jj}|, j= 1,\ldots,n \} \cdot \varepsilon_M$$

where ε_M is the computer rounding unit.

(2) We must observe that pivoting permits the factorization to progress until all remaining diagonal elements are null or negative. This will be useful for our algorithm of quadratic programming, but in order to prevent numerical instability it is necessary to control the growth of the Cholesky factors. So we have incorporated a parameter β to our decomposition algorithm which controls the growth of these factors. Indeed it is easy to verify that the realized choice of β implies that if

$$|a_{ik}^{(k+1)}| > \beta \qquad \text{for some i from k+1 to n}$$

then the diagonal element $a_{ii}^{(k+1)}$ would be negative if it were computed.

(3) If the algorithm is stopped in the iteration k with an indication of indefiniteness, then we have got a permutation matrix $P = P^{(k)}$ and the following factorization of PAP^T

$$PAP^T = \begin{pmatrix} L & 0 \\ B & I_{n-m} \end{pmatrix} \begin{pmatrix} I_m & 0 \\ 0 & C \end{pmatrix} \begin{pmatrix} L^T & B^T \\ 0 & I_{n-m} \end{pmatrix}$$

where m = k-1, L is a m x m non singular lower-triangular matrix, B is (n-m) x

m and C is a (n-m) x (n-m) negative semi-definite symmetric matrix having some strictly negative curvature directions. These matrices are defined by the equalities

$$l_{ij} = a_{ij}^{(k)} \qquad 1 \le j \le i \le m$$

$$c_{ij} = a_{l+m,j+m}^{(k)} \qquad 1 \le j \le i \le n-m$$

$$b_{ij} = a_{m+l,j}^{(k)} \qquad 1 \le i \le n-m, \quad 1 \le j \le m$$

From this factorization we can obtain some negative curvature directions. Firstly let us suppose that $a_{jj}^{(k)} < 0$ for some index $j \ge k$, then the n-vector $P^T u$, with u defined by the equality

$$u = \begin{pmatrix} \hat{u} \\ 0 \end{pmatrix} + e_j$$

where $L^T\hat{u} = -B_{j-m}^T$ and B_{j-m} is the (j-m)-th row of B, is a negative curvature direction of A:

$$(P^T u)^T A (P^T u) = \left(\begin{pmatrix} \hat{u} \\ 0 \end{pmatrix} + e_j \right)^T \begin{pmatrix} L & 0 \\ B & I_{n-m} \end{pmatrix} \begin{pmatrix} I_m & 0 \\ 0 & C \end{pmatrix} \begin{pmatrix} L^T & B^T \\ 0 & I_{n-m} \end{pmatrix} \left(\begin{pmatrix} \hat{u} \\ 0 \end{pmatrix} + e_j \right) =$$

$$\left(\begin{pmatrix} L^T\hat{u} + B_{j-m}^T \\ 0 \end{pmatrix} + e_j \right)^T \begin{pmatrix} I_m & 0 \\ 0 & C \end{pmatrix} \left(\begin{pmatrix} L^T\hat{u} + B_{j-m}^T \\ 0 \end{pmatrix} + e_j \right) =$$

$$e_j^T \begin{pmatrix} I_m & 0 \\ 0 & C \end{pmatrix} e_j = a_{jj}^{(k)} < 0$$

Now we assume that $a_{jj}^{(k)} = 0$ for $j = k,\ldots,n$ and $|a_{rs}^{(k)}| > 0$ for some r,s, with $k \leq s < r \leq n$. In this case we take the n-vector

$$u = \begin{pmatrix} \hat{u} \\ 0 \end{pmatrix} - a_{rs}^{(k)} e_s + e_r$$

where $L^T\hat{u} = -B_{r-m}^T + a_{rs}^{(k)} B_{s-m}^T$. So $P^T u$ is a negative curvature vector of A:

$$(P^T u)^T A (P^T u) =$$

$$\left(\begin{pmatrix} \hat{u} \\ 0 \end{pmatrix} - a_{rs}^{(k)} e_s + e_r \right)^T \begin{pmatrix} L & 0 \\ B & I_{n-m} \end{pmatrix} \begin{pmatrix} I_m & 0 \\ 0 & C \end{pmatrix} \begin{pmatrix} L^T & B^T \\ 0 & I_{n-m} \end{pmatrix} \left(\begin{pmatrix} \hat{u} \\ 0 \end{pmatrix} - a_{rs}^{(k)} e_s + e_r \right)$$

$$= (-a_{rs}^{(k)} e_s + e_r)^T \begin{pmatrix} I_m & 0 \\ 0 & C \end{pmatrix} (-a_{rs}^{(k)} e_s + e_r) =$$

$$(a_{rs}^{(k)})^2 c_{s-m,s-m} + c_{r-m,r-m} - 2a_{rs}^{(k)} c_{r-m,s-m} =$$

$$(a_{rs}^{(k)})^2 a_{s,s}^{(k)} + a_{r,r}^{(k)} - 2(a_{rs}^{(k)})^2 < 0$$

In any case, we have seen that it is not numerically difficult to get negative curvature directions of A from the partial Cholesky factorization furnished by the above algorithm. If the matrix A is positive semi-definite Theorem 1 supplies a procedure for obtaining a basis of the kernel of A. In both cases the method is based on the resolution of a system of linear equations with a coefficient matrix L^T that is upper-triangular.

3. THE MODEL ALGORITHM FOR QUADRATIC PROGRAMMING

As usual we consider an iterative procedure that follows the active set strategy for solving quadratic programming problems. So in each iteration we have a feasible point x^k, a $n \times m_k$ matrix C_k whose i-th column contains the coefficients of the i-th active constraint, m_k being the number of active constraints in iteration k. We will denote by I_k the index set corresponding to the active constrains.

Following Gill and Murray [6], we factorize the matrix C_k into the product $Q_k R_k$, where Q_k is an orthogonal matrix and R_k is an upper-triangular matrix. In Q_k and R_k we distinguish the submatrices:

$$Q_k = [Y_k \; S_k] \qquad \text{and} \qquad R_k = \begin{pmatrix} \hat{R}_k \\ 0 \end{pmatrix}$$

where Y_k is $n \times m_k$, S_k is $n \times (n-m_k)$ and $\hat{R}_k$ is $m_k \times m_k$. The columns of Y_k form an orthonormal basis for the range space of C_k and those of S_k form an orthonormal basis of the null space of C_k^T. Finally we get the reduced Hessian H_k and compute its Cholesky decomposition in the way described in the previous section (remark that H_k is symmetric).

The reduced Hessian could be computed by the formula $H_k = S_k^T H S_k$, but we prefer to define $H_k = Z_k^T H Z_k$ by reasons that we will explain in Section 4, where Z_k is the matrix obtained from S_k by setting its columns in reverse order , that is to say $Z_k = S_k \tilde{I}$, $\tilde{I}$ being the matrix of order $n-m_k$:

$$\tilde{I} = \begin{pmatrix} 0 & & 1 \\ & \iddots & \\ 1 & & 0 \end{pmatrix}$$

It is well known that there are other ways to get a basis of the null space of C_k^T, mainly based on the use of LU factorization, see Fletcher [4]. Here we prefer the QR factorization because of its good properties of numerical stability, see Wilkinson [9], and because we are assuming that the matrices H and C are not sparse and they can be stored explicitly in the main memory of the computer. For large sparse matrices the LU factorization might be preferable.

Our quadratic programming algorithm performs the following steps:

1) Compute a feasible initial point x^0 and set $k = 0$.
Compute the factorization QR of the matrix C_k of active constraints, get Z_k and the reduced Hessian H_k.
Apply the Cholesky decomposition algorithm to H_k as above indicated.

2) If H_k is not positive semi-definite or $Z_k^T \nabla F(x^k) \neq 0 \Rightarrow$ go to 3.
Otherwise compute the Lagrange multipliers λ^k through the equation

$$\hat{R}_k \lambda = -Y_k^T \nabla F(x^k)$$

If all Lagrange multipliers associated to the active inequality constraints are positive then we have found a local solution of the problem. STOP. Otherwise we remove the inequality constraint

corresponding to the smallest negative Lagrange multiplier and modify the QR factors of the active constraint matrix and the Cholesky decomposition of the reduced Hessian. Go to Step 3).

3) Compute a descent direction:

If H_k is positive definite, then solve the system

$$H_k d_{Z_k} = -Z_k^T \nabla F(x^k)$$

using the Cholesky factorization and take $d^k = Z_k d_{Z_k}$. Go to Step 4).

If H_k is positive semi-definite, then compute a basis $\{u_j\}_{j=1}^{n-m_k}$ of the null space of H_k from the formulas given in Theorem 1 and take

$$\hat{d}^k = -Z_k U_k U_k^T Z_k^T \nabla F(x^k)$$

where U_k is the matrix whose columns are the vectors u_j.

If $\hat{d}^k \neq 0 \Rightarrow d^k = \hat{d}^k$ and go to Step 5).

If $\hat{d}^k = 0 \Rightarrow$ Solve the system

$$H_k d_{Z_k} = -Z_k^T \nabla F(x^k)$$

and take $d^k = Z_k d_{Z_k}$ and go to Step 4).

If H_k is not positive semi-definite, then compute a descent direction of negative curvature and continue at Step 5).

4) Compute ρ_k from the formula:

$$\rho_k = \min \{ 1, \min_{j \notin I_k,\ c_j^T d^k > 0} \frac{b_j - c_j^T x^k}{c_j^T d^k} \}$$

Take $x^{k+1} = x^k + \rho_k d^k$ and $k \to k+1$.

If $\rho_k = 1$ then go to Step 2), otherwise continue at Step 6).

5) Compute ρ_k from the formula:

$$\rho_k = \min_{j \notin I_k,\ c_j^T d^k > 0} \frac{b_j - c_j^T x^k}{c_j^T d^k}$$

If the index set where we look for the minimum is empty, then the quadratic problem is unbounded below in the feasible region, so there is no solution. STOP.

Otherwise take $x^{k+1} = x^k + \rho_k d^k$ and $k \to k+1$. Go to Step 6).

6) If ρ_k is the step corresponding to the constraint with index j_k, add j_k to I_k and c_{j_k} to the matrix C_k. Modify the QR factorization and Cholesky decomposition of the new reduced Hessian. Go back to Step 2).

REMARKS. (1) First we must remark that neither the initial point x^0 need to be a vertex nor we need to introduce any artificial constraint. The Cholesky algorithm proposed in Section 2 allows us to compute a descent direction d^k in a stable way beginning with any symmetric matrix H_k. Among the possible descent directions computed by our quadratic programming algorithm, we find null, positive or negative curvature directions.

(2) In practice the determination of $Z_k^T \nabla F(x^k)$ and $\hat{d}^k$, in Step 2) and 3) of our algorithm, as null vectors is based on the comparison of its norm with a small parameter depending on machine precision.

(3) Our code deletes inequality constraints with zero Lagrangre multipliers. In Step 2), if the smallest Lagrange multiplier associated with an active inequality constraint is null then the corresponding constraint is removed from the working set. If the new reduced Hessian has a negative eigenvalue, then, a descent direction of negative curvature is computed; else the new reduced Hessian is positive semi-definite and the following null Lagrange multiplier is studied. So we can infer the theorem

THEOREM 2. If the objective function is bounded below in the feasible region and there exist no degenerated stationary points in this region, the previous algorithm converges in a finite number of iterations to a local minimum x.

PROOF. Before removing an active constraint of the working set, the algorithm has found the minimum of the current equality constrained problem. Therefore, after a finite number of iterations the algorithm must find a point x and a set of active constraints where the reduced Hessian associated is positive semi-definite, the reduced gradient is null and the Lagrange multipliers corresponding to the inequality constraints of the working set are strictly positives. Under these conditions is well known that this point x is a local minimum, see for example Fletcher [4].

It is important to remark here the necessity of the Lagrange multipliers associated to inequality constraints to be strictly positive if we want to be sure that x is a local minimum. Indeed, let us consider the following example:

$$\text{Minimize} \qquad F(x) = x_3^2 - 2x_1x_2$$

$$\text{Subject to} \qquad 0 \leq x_1 + x_2 \leq 2$$

$$x_1 - x_2 \leq -2$$

Let be $x = (-1,1,0)^T$ and $\lambda = (0,2)^T$, then (x,λ) is a Kuhn-Tucker point and the reduced Hessian

$$Z^THZ = (0,0,1)\begin{pmatrix} 0 & -2 & 0 \\ -2 & 0 & 0 \\ 0 & 0 & 2 \end{pmatrix}\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = 2$$

is positive definite. However x is not a local minimum because $F(x_\varepsilon) < F(x)$ for each $\varepsilon \neq 0$, where $x_\varepsilon = (\varepsilon-1,\varepsilon+1,0)^T$: $F(x_\varepsilon) = 2(1-\varepsilon^2) < 2 = F(x)$.

(4) In Step 3) of our algorithm, there are two different ways to compute the descent direction in the case of a reduced Hessian that is positive semi-definite. First the projection of the reduced gradient on the null space of the reduced Hessian is computed. If this projection is null, it is not possible to get a descent direction of null curvature, however, in this situation, the system

$$H_k d_{z_k} = -Z_k^T \nabla F(x^k)$$

has at least one solution (in fact it has many solutions) because of the orthogonality of $Z_k^T \nabla F(x^k)$ on the null space of H_k. In this way we obtain a descent direction of positive curvature, which allows us to solve the above example. When $Z_k^T \nabla F(x^k)$ is not orthogonal to the kernel of H_k, then the above system has not any solution, but fortunately we can compute a descent direction of null curvature in this case such as it was pointed out in the algorithm.

The solution of the above system, when it exists, is computed in the following way. Firstly we have computed the Cholesky factorization of the reduced Hessian (see Theorem 1)

$$H_k = P_k^T \begin{pmatrix} L_k & 0 \\ B_k & 0 \end{pmatrix} \begin{pmatrix} L_k^T & B_k^T \\ 0 & 0 \end{pmatrix} P_k$$

Now we denote by v^k the vector formed by the m_k first components of the vector $-P_k Z_k^T \nabla F(x^k)$ and compute w^k as solution of the system

$$L_k L_k^T w^k = v^k$$

Finally we take

$$d_{Z_k} = P_k^T \begin{pmatrix} w^k \\ 0 \end{pmatrix}$$

Let us verify that this vector is a solution of the above system:

$$H_k d_{Z_k} = P_k^T \begin{pmatrix} L_k & 0 \\ B_k & 0 \end{pmatrix} \begin{pmatrix} L_k^T & B_k^T \\ 0 & 0 \end{pmatrix} P_k \; P_k^T \begin{pmatrix} w^k \\ 0 \end{pmatrix} =$$

$$P_k^T \begin{pmatrix} L_k & 0 \\ B_k & 0 \end{pmatrix} \begin{pmatrix} L_k^T & B_k^T \\ 0 & 0 \end{pmatrix} \begin{pmatrix} w^k \\ 0 \end{pmatrix} = P_k^T \begin{pmatrix} L_k L_k^T w^k \\ B_k L_k^T w^k \end{pmatrix} = P_k^T \begin{pmatrix} v^k \\ B_k L_k^T w^k \end{pmatrix}$$

Let us denote by b^k the $n-m_k$ last components of $-P_k Z_k^T \nabla F(x^k)$ and for every index j, with $1 \le j \le n-m_k$, let $P_k^T u_j$ be the vector of null space of H_k defined by (Theorem 1)

$$u_j = \begin{bmatrix} \hat{u}_j \\ 0 \end{bmatrix} - e_{m_k+j}$$

with $L_k^T \hat{u}_j = (B_k)_j^T$.

Because $Z_k^T \nabla F(x^k)$ is orthogonal to each vector $P_k^T u_j$ we have

$$0 = \left(P_k^T u_j \right)^T \left(-Z_k^T \nabla F(x^k) \right) = - u_j^T P_k Z_k^T \nabla F(x^k) = u_j^T \begin{pmatrix} v^k \\ b^k \end{pmatrix} = \hat{u}_j^T v^k - b_j^k$$

So $\hat{u}_j^T v^k = b_j^k$ and therefore

$$\left(B_k L_k^T w^k \right)_j = \left(B_k \right)_j L_k^T w^k = \left(L_k^T \hat{u}_j \right)^T L_k^T w^k = \hat{u}_j^T \left(L_k L_k^T w^k \right) = \hat{u}_j^T v^k = b_j^k$$

Hence we conclude

$$H_k d_{Z_k} = P_k^T \begin{pmatrix} v^k \\ B_k L_k^T w^k \end{pmatrix} = P_k^T \begin{pmatrix} v^k \\ b^k \end{pmatrix} = -P_k^T P_k Z_k^T \nabla F(x^k) = -Z_k^T \nabla F(x^k)$$

4. MODIFICATION OF QR AND CHOLESKY FACTORS

It is well known that as changes are produced in the active set, the QR and Cholesky factorizations can be modified rather than be computed ab initio, see Gill et al. [5]. We must distinguish two cases because these changes are a consequence of adding or removing a constraint from the active set.

4.1 DELETING A CONSTRAINT

Let C be the n x m active constraint matrix and C = QR, Q = (Y S) and

$$R = \begin{pmatrix} \hat{R} \\ 0 \end{pmatrix}$$

Let us assume that the l-th constraint is deleted from the working set and let $\bar{C}$ be the new active constraint matrix. Let us denote by $\bar{R}$ the matrix obtained from R removing the l-th column, then $\bar{C} = Q\bar{R}$. In order to reduce $\bar{R}$ to the triangular form it is enough to apply m-l orthogonal transformations (for example Householder or Givens transformations) on the last columns of $\bar{R}$ and to perform the corresponding modifications on Q. The modifications on Q do not affect the last n-m columns, hence the new orthogonal matrix will be $\bar{Q} = (\bar{Y}\ \bar{S})$ with S being augmented in a column and therefore $\bar{Z} = (Z\ z)$.

If we denote by $H_z = Z^T H Z$ the reduced Hessian before removing the l-th

constraint, the new Hessian will be:

$$H_{\bar{z}} = \bar{Z}^T H \bar{Z} = \begin{pmatrix} Z^T \\ z^T \end{pmatrix} H \, (Z \; z) = \begin{pmatrix} Z^T HZ & Z^T Hz \\ z^T HZ & z^T Hz \end{pmatrix}$$

Since we are deleting a constraint H_z must be positive semi-definite, so we know its Cholesky decomposition:

$$H_z = P^T \begin{pmatrix} L & 0 \\ B & 0 \end{pmatrix} \begin{pmatrix} L^T & B^T \\ 0 & 0 \end{pmatrix} P$$

From here it follows that

$$H_{\bar{z}} = \begin{pmatrix} P & 0 \\ 0 & 1 \end{pmatrix}^T \begin{pmatrix} \begin{pmatrix} L & 0 \\ B & 0 \end{pmatrix} \begin{pmatrix} L^T & B^T \\ 0 & 0 \end{pmatrix} & PZ^T Hz \\ z^T HZP^T & z^T Hz \end{pmatrix} \begin{pmatrix} P & 0 \\ 0 & 1 \end{pmatrix}$$

Let a_1 be the vector formed by the m first components of $PZ^T Hz$ and a_2 the n-m vector formed by the last components. Now we take b_1 satisfying $Lb_1 = a_1$ and we distinguish two cases:

First Case: $z^T Hz - b_1^T b_1 > 0$

In this case we take $\bar{b} = \sqrt{z^T Hz - b_1^T b_1}$ and $b_2 = \dfrac{a_2 - Bb_1}{\bar{b}}$ and then we have

$$H_{\bar{z}} = \begin{pmatrix} P & 0 \\ 0 & 1 \end{pmatrix}^T \begin{pmatrix} L & 0 & 0 \\ B & b_2 & I_B \\ b_1^T & \bar{b} & 0 \end{pmatrix} \begin{pmatrix} I_L & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -b_2 b_2^T \end{pmatrix} \begin{pmatrix} L^T & B^T & b_1 \\ 0 & b_2^T & \bar{b} \\ 0 & I_B & 0 \end{pmatrix} \begin{pmatrix} P & 0 \\ 0 & 1 \end{pmatrix}$$

where I_B and I_L denote the identity matrices of the same order than B and L respectively. Now changing the order of the last row or column we get

$$H_{\bar{z}} = \bar{P}^T \begin{pmatrix} L & 0 & 0 \\ b_1^T & \bar{b} & 0 \\ B & b_2 & I_B \end{pmatrix} \begin{pmatrix} I_L & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -b_2 b_2^T \end{pmatrix} \begin{pmatrix} L^T & b_1 & B^T \\ 0 & \bar{b} & b_2^T \\ 0 & 0 & I_B \end{pmatrix} \bar{P} =$$

$$\bar{P}^T \begin{pmatrix} \bar{L} & 0 \\ \bar{B} & I_{\bar{B}} \end{pmatrix} \begin{pmatrix} I_{\bar{L}} & 0 \\ 0 & -b_2 b_2^T \end{pmatrix} \begin{pmatrix} \bar{L}^T & \bar{B}^T \\ 0 & I_{\bar{B}} \end{pmatrix} \bar{P}$$

where $\bar{P}$ is the matrix obtained from

$$\begin{pmatrix} P & 0 \\ 0 & 1 \end{pmatrix}$$

pivoting the corresponding row.

If $b_2 \neq 0$ then $H_{\bar{z}}$ has a negative curvature direction that can be computed as indicated in Section 2 from the above decomposition. When $b_2 = 0$ then $H_{\bar{z}}$ is positive semidefinite and its Cholesky decomposition is

$$H_{\bar{z}} = \bar{P}^T \begin{pmatrix} \bar{L} & 0 \\ \bar{B} & 0 \end{pmatrix} \begin{pmatrix} \bar{L}^T & \bar{B}^T \\ 0 & 0 \end{pmatrix} \bar{P}$$

Second Case: $z^T Hz - b_1 b_1^T \leq 0$

In this case we take $\bar{b} = z^T Hz - b_1^T b_1$ and $u = a_2 - Bb_1$ and so we get

$$H_{\bar{Z}} = \begin{pmatrix} P & 0 \\ 0 & 1 \end{pmatrix}^T \begin{pmatrix} L & 0 & 0 \\ B & I_B & 0 \\ b_1^T & 0 & 1 \end{pmatrix} \begin{pmatrix} I_L & 0 & 0 \\ 0 & 0 & u \\ 0 & u^T & \bar{b} \end{pmatrix} \begin{pmatrix} L^T & B^T & b_1 \\ 0 & I_B & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} P & 0 \\ 0 & 1 \end{pmatrix} =$$

$$\bar{P}^T \begin{pmatrix} \bar{L} & 0 \\ \bar{B} & I_{\bar{B}} \end{pmatrix} \begin{pmatrix} I_L & 0 & 0 \\ 0 & 0 & u \\ 0 & u^T & \bar{b} \end{pmatrix} \begin{pmatrix} \bar{L}^T & \bar{B}^T \\ 0 & I_{\bar{B}} \end{pmatrix} \bar{P}$$

where $\bar{L} = L$ and $\bar{P} = P$. The matrix $H_{\bar{Z}}$ has a negative curvature direction if $\bar{b} \neq 0$ or $u \neq 0$, otherwise it would be positive semi-definite with a Cholesky decomposition associated to the matrices $\bar{L}$ and $\bar{B}$.

If $u \neq 0$ or $\bar{b} \neq 0$ we can get a negative curvature direction for $H_{\bar{Z}}$ in the following way

$$L^T\hat{u} = -b_1 + B^T u \qquad \text{and} \qquad d_{\bar{Z}} = \begin{pmatrix} \hat{u} \\ -u \\ 1 \end{pmatrix}$$

Now it is easy to verify that $(\bar{P}^T d_{\bar{Z}})^T H_{\bar{Z}} \bar{P}^T d_{\bar{Z}} = -2u^T u + \bar{b} < 0$.

4.2 ADDING A CONSTRAINT

Let C, Q = (Y S), R and Z be as in previous section and assume that the constraint associated to the coefficient vector c is to be added to the active set. So we define the new matrix of constraints active $\bar{C}$ = (C c) and we have the equality $\bar{C}$ = Q(R Q^Tc). Hence in order to derive the QR factorization of $\bar{C}$ it is enough to find an orthogonal transformation that vanish the n-m-1 last

components of $Q^T c$. This transformation can be a Housholder matrix or a product of Givens rotations. Since we have to compute the new Cholesky factors of H_z, it is preferable to apply Givens rotations in a certain order as we are going to see. For the moment let M be a product of n-m-1 Givens matrices of plane rotation that turn $(R\ Q^T c)$ into an upper-triangular matrix. M affects only the last n-m rows of $(R\ Q^T c)$ and its form is

$$M = \begin{pmatrix} I_m & 0 \\ 0 & \hat{M} \end{pmatrix}$$

with $\hat{M}$ orthogonal matrix product of plane rotations, so $\bar{Q} = QM^T = (Y\ S\hat{M}^T)$ and

$$\bar{R} = \begin{pmatrix} \hat{R} & u \\ 0 & \hat{M}v \end{pmatrix}$$

where u represents the m first components of $Q^T c$ and v the n-m last, $\hat{M}v$ having null all its components, except the first.

Take now $\hat{Z} = S\hat{M}^T\tilde{I} = (\bar{Z}\ z)$. It is immediate to verify that $\bar{Z}$ is a matrix whose columns are orthogonal and form a basis of the null space of $\bar{C}^T$. We want to obtain the Cholesky factors of $H_{\bar{z}} = \bar{Z}^T H \bar{Z}$. For it we use the Cholesky decomposition of H_z (see Section 2):

$$H_z = P^T \begin{pmatrix} L & 0 \\ B & I_B \end{pmatrix} \begin{pmatrix} I_L & 0 \\ 0 & D \end{pmatrix} \begin{pmatrix} L^T & B^T \\ 0 & I_B \end{pmatrix} P$$

where D is the null matrix if H_z is positive semi-definite and D is negative semi-definite if H_z is indefinite. If H_z is positive definite the previous

factorization is reduced to $H_Z = P^T L L^T P$.

Let us define $\hat{H}_Z = \hat{Z}^T H \hat{Z}$. Then, from equality $Z = S\tilde{I}$, it follows

$$\hat{H}_Z = \hat{Z}^T H \hat{Z} = \tilde{I}\hat{M}S^T H S \hat{M}^T \tilde{I} = \tilde{I}\hat{M}\tilde{I}Z^T H Z \tilde{I}\hat{M}^T\tilde{I} = \tilde{I}\hat{M}\tilde{I}H_Z\tilde{I}\hat{M}^T\tilde{I} =$$

$$\tilde{I}\hat{M}\tilde{I}P^T \begin{pmatrix} L & 0 \\ B & I_B \end{pmatrix} \begin{pmatrix} I_L & 0 \\ 0 & D \end{pmatrix} \begin{pmatrix} L^T & B^T \\ 0 & I_B \end{pmatrix} P\tilde{I}\hat{M}^T\tilde{I}$$

If P = Identity and the Givens rotations are applied in the planes (n,n-1), (n-1,n-2),..., (m+2,m+1), then the matrix

$$X = \tilde{I}\hat{M}\tilde{I}P^T \begin{pmatrix} L & 0 \\ B & I_B \end{pmatrix}$$

is lower-Hessenberg of the form (see Gill and Murray [6])

$$X = \begin{pmatrix} r & N \\ \sigma & s^T \end{pmatrix} \quad \text{with} \quad N = \begin{pmatrix} N_{11} & 0 \\ N_{21} & N_{22} \end{pmatrix} \quad \text{and} \quad r = \begin{pmatrix} r_1 \\ r_2 \end{pmatrix}$$

When P is a pivoting matrix different of the identity we can reach the same result by performing the Givens rotations in a different order. This order is the following: $\hat{M} = \hat{M}_{i_k,j_k} \cdots \hat{M}_{i_1,j_1}$, where $k = n-m-1$ and $\hat{M}_{i_q,j_q}$ is a rotation in the plane $(n+1-i_q, n+1-j_q)$, $q = 1,\ldots,k$. The pairs $(n+1-i_1, n+1-j_1)$, $\ldots,(n+1-i_k, n+1-j_k)$ are formed from the vector JPVT that indicates the pivotations realized for the Cholesky factorization, in other words it is a vector associated to the matrix P. P is a permutation matrix obtained from the

identity matrix permuting its rows and JPVT(q), $q = 1,\ldots,n-m$, contains the index of the row of the identity matrix that was moved into the q-th position. Now we form the pairs (i_q, j_q) in the following way:

If JPVT(1) > JPVT(2) $\Rightarrow i_1 =$ JPVT(1) and $j_1 =$ JPVT(2)

Else $i_1 =$ JPVT(2) and $j_1 =$ JPVT(1).

From q = 2 to n-m

If $i_{q-1} >$ JPVT(q) $\Rightarrow i_q = i_{q-1}$ and $j_q =$ JPVT(q)

Else $i_q =$ JPVT(q) and $j_q = i_{q-1}$

End

So we have

$$\hat{H}_z = X \begin{pmatrix} I_L & 0 \\ 0 & D \end{pmatrix} X^T = \begin{pmatrix} rr^T + N\hat{D}N^T & \sigma r + N\hat{D}s \\ \sigma r^T + s^T\hat{D}N^T & \sigma^2 + s^T\hat{D}s \end{pmatrix}$$

where $\hat{D}$ is the matrix that satisfies

$$\begin{pmatrix} I_L & 0 \\ 0 & D \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & \hat{D} \end{pmatrix}$$

On the other hand

$$\hat{H}_z = \hat{Z}^T H \hat{Z} = (\bar{Z}\ z)^T H\ (\bar{Z}\ z) = \begin{pmatrix} \bar{Z}^T H\bar{Z} & \bar{Z}^T Hz \\ z^T H\bar{Z} & z^T Hz \end{pmatrix} = \begin{pmatrix} H_{\bar{z}} & \bar{Z}^T Hz \\ z^T H\bar{Z} & z^T Hz \end{pmatrix}$$

Finally we get

$$H_{\underline{Z}} = rr^T + \hat{N}DN^T = \begin{pmatrix} N_{11} & 0 & r_1 \\ N_{21} & N_{22} & r_2 \end{pmatrix} \begin{pmatrix} I_{11} & 0 & 0 \\ 0 & D & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} N_{11}^T & N_{21}^T \\ 0 & N_{22}^T \\ r_1^T & r_2^T \end{pmatrix} =$$

$$\begin{pmatrix} N_{11} & r_1 & 0 \\ N_{21} & r_2 & N_{22} \end{pmatrix} \begin{pmatrix} I_{11} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & D \end{pmatrix} \begin{pmatrix} N_{11}^T & N_{21}^T \\ r_1^T & r_2^T \\ 0 & N_{22}^T \end{pmatrix}$$

where I_{11} is the identity matrix of the same order than N_{11}. Using again Givens transformations we obtain an orthogonal matrix G such that

$$G \begin{pmatrix} N_{11}^T \\ r_1^T \end{pmatrix} = \begin{pmatrix} \bar{N}_{11}^T \\ 0 \end{pmatrix}$$

$\bar{N}_{11}$ being a lower-triangular matrix. Now we introduce the following notation

$$\begin{pmatrix} \bar{N}_{11} & 0 \\ \bar{N}_{21} & \bar{r}_2 \end{pmatrix} = \begin{pmatrix} N_{11} & r_1 \\ N_{21} & r_2 \end{pmatrix} G^T$$

and

$$\bar{G} = \begin{pmatrix} G & 0 \\ 0 & I_{22} \end{pmatrix}$$

with I_{22} equal to identity matrix that has the same order than N_{22}. From the

factorization above obtained for $H_{\bar{Z}}$ we deduce

$$H_{\bar{Z}} = \begin{pmatrix} \bar{N}_{11} & 0 & 0 \\ \bar{N}_{21} & \bar{r}_2 & N_{22} \end{pmatrix} \bar{G} \begin{pmatrix} I_{11} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & D \end{pmatrix} \bar{G}^T \begin{pmatrix} \bar{N}_{11}^T & \bar{N}_{21}^T \\ 0 & \bar{r}_2^T \\ 0 & \bar{N}_{22}^T \end{pmatrix} =$$

$$\begin{pmatrix} \bar{N}_{11} & 0 \\ \bar{N}_{21} & I_{21} \end{pmatrix} \begin{pmatrix} I_{11} & 0 \\ 0 & r_2 r_2^T + N_{22} D N_{22}^T \end{pmatrix} \begin{pmatrix} \bar{N}_{11}^T & \bar{N}_{21}^T \\ 0 & I_{21} \end{pmatrix}$$

Finally we obtain a Cholesky factorization for $H_{\bar{Z}}$ similar to that of H_Z by performing the decomposition of the matrix $r_2 r_2^T + N_{22} D N_{22}^T$. In this way we arrive to the final descomposition

$$H_{\bar{Z}} = \bar{P}^T \begin{pmatrix} \bar{L} & 0 \\ \bar{B} & I_{\bar{B}} \end{pmatrix} \begin{pmatrix} I_{\bar{L}} & 0 \\ 0 & \bar{D} \end{pmatrix} \begin{pmatrix} \bar{L}^T & \bar{B}^T \\ 0 & I_{\bar{B}} \end{pmatrix} \bar{P}$$

where the order of matrix $\bar{D}$ is the same or one unity lower to that of D.

In practice in order to prevent numerical instability, $r_2 r_2^T + N_{22} D N_{22}^T$ is not computed and the factorization of this matrix is obtained from the corresponding submatrix of $\bar{Z}^T H \bar{Z}$ following the algorithm described in Section 2.

5. NUMERICAL EXAMPLES

Our algorithm has been implemented in a FORTRAN 77 program and applied to

several examples. Many of these examples were generated by using the random function of the machine, but forcing sometimes the matrix H to be singular or positive definite. Our code finished successfully in all cases detecting a function not bounded from below or finding a local minimum. Here we present two examples. In the first example the matrix H is singular and positive semi-definite. This is a difficult problem as shown by the fact that the routine E04NAF of the NAG Library failed to get the solution. The second example was constructed by J.R. Bunch and L. Kaufman [1]. In this case the matrix H has two negative eigenvalues.

EXAMPLE 1

$$\text{Minimize} \quad \frac{1}{2}(x_1,x_2,x_3,x_4)\begin{pmatrix} 4 & -2 & 2 & 2 \\ -2 & 2 & 2 & 1 \\ 2 & 2 & 10 & 7 \\ 2 & 1 & 7 & 5 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} + (2,-2,-2,-1)\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix}$$

$$\text{Subject to} \quad x_2 + 3x_3 + 2x_4 = 0$$

$$2x_1 - x_2 + x_3 + x_4 \leq 0$$

The set of solutions of this problem is the following:

$$\begin{pmatrix} -4 \\ -5 \\ 1 \\ 1 \end{pmatrix} + \alpha\begin{pmatrix} 2 \\ 3 \\ -1 \\ 0 \end{pmatrix} + \beta\begin{pmatrix} 3 \\ 4 \\ 0 \\ -2 \end{pmatrix}$$

where α,β are any real numbers. Our algorithm found a solution in two iterations.

EXAMPLE 2.

Minimize $F(x) = \frac{1}{2}x^T Hx + p^T x$

where

$$h_{ij} = \begin{cases} |i-j| & \text{if } i \neq j, \\ 1.69 & \text{if } i=j, \end{cases} \qquad \text{and} \qquad p = (7,6,\ldots,0)^T$$

subject to the bound constraints:

$$-i-(i-1)x0.1 \leq x_i \leq i, \qquad i = 1,2,\ldots,8$$

and the inequality constraints:

$$x_i - x_{i+1} \leq 1 + (i-1)x0.05, \qquad i = 1,2,\ldots,7$$

The problem has a local minimum of F=-621.487825 at the point

$$x^T = (-1,-2,-3.05,-4.15,-5.3,6,7,8)$$

one of -642.643525 at

$$x^T = (-1,-2.1,-3,15,-4.25,-5.4,6,7,8)$$

and one of -131.774167868... at

$$x^T = (1,2,1.880144,0.780144,-0.369856,-1.569856,-2.819856,-4.119856)$$

Begining at $x_i = -i$, the routine reached the second local minimum at 7

iterations. The course of the algorithm was the following:

ITERATION 1:

Active Constraints: -1,9

A descent direction of negative curvature was computed.

Added Constraint: 10

ITERATION 2:

Active Constraints: -1,9,10

A descent direction of negative curvature was computed.

Added Constraint: 11

ITERATION 3:

Active constraints: -1,9,10,11

A descent direction of negative curvature was computed.

Added Constraint: 12

ITERATION 4:

Active constraints: -1,9,10,11,12

A descent direction of negative curvature was computed.

Added Constraint: 13

ITERATION 5:

Active constraints: -1,9,10,11,12,13

A descent direction of positive curvature was computed.

Added Constraint: 8

ITERATION 6:

Active constraints: -1,9,10,11,12,13,8

A descent direction of positive curvature was computed.

Added Constraint: 7

ITERATION 7:

Active constraints: -1,9,10,11,12,13,8,7

Deleted Constraint: 13

A descent direction of positive curvature was computed.

Added Constraint: 6

End of Routine: A local minimum was founded.

The convention for numbering the consttraints is the following:

i<0 : lower bound -i.

1≤i≤n : upper bound i.

i>n : inequality constraint i-n.

REFERENCES

[1] J.R. Bunch and L. Kaufman: "A computational method for the indefinite quadratic programming problem", Linear Algenra and its Applications, 34, 341-370, 1980.

[2] J.J. Dongarra, C.B. Moler, J.R. Bunch and G.W. Stewart: "LINPACK user's guide". Ed. SIAM. Philadelphia. 1979.

[3] R. Fletcher: "A general quadratic programming algorithm". J. Institue of Mathematics and its Applications, 7, 76-91, 1971.

[4] R. Fletcher: "Practical methods of optimization". Vol. 2. Ed. John Wiley & Sons. New York. 1981.

[5] P.E. Gill, G.H. Golub, W. Murray and M.A. Saunders: "Methods for modifying matrix factorizations". Math. of Comp. 28 (126), 505-535. 1974.

[6] P.E. Gill and W. Murray: "Numerically stable methods for quadratic programming". Math. Programming 14, 349-372. 1978.

[7] P.E. Gill, W. Murray and M.H. Wright: "Practical optimization". Ed. Academic Press. London. 1981.

[8] P.E. Gill, W. Murray, M.A. Saunders and M.H. Wright: "Inertia-controlling methods for quadratic programming", Report SOL 88-3, Department of Operations Research, Stanford University, 1988.

[9] J.H. Wilkinson: "The algebraic eigenvalue problem". Ed. Oxford University Press. Oxford. 1965.

Imprimé en France
par
l'Institut National de Recherche en Informatique et en Automatique